BIOINFO Security Informatics
Volume 1, Issue 1, 2011, pp-01-05
Available online at: http://www.bioinfo.in/contents.php?id=344

Dynamic Security Skins-Mutual Authentication

V.P. Surwase and S.R. Durugkar
[1]*Department of C.S., S.N.D.C.O.E & R.C., Yeola, Nasik*
[2]*Department of I.T., Sinhgad College of Engineering, Vadgaon, Pune*
e-mail: surwase.vaishnav@gmail.com, santoshdurugkar@gmail.com

Abstract—Phishing means, attack combines Social engineering & technology; an attack that tricks users out of confidential information/ credentials such as; Authentication, Financial. In fact Phishing is of spoofing personal credential.

This paper presents a scheme, Dynamic Security Skins, that allows a remote web server to prove its identity in a way that is easy for a human user to verify & hard for an attacker to spoof. These schemes include a new password authentication & key-exchange protocol suitable for authenticating users & exchanging keys over an un-trusted network. In short DSS minimize user memory requirements. As the final result the DSS provide toughest security than current SSL based schemes without altering user approach, characteristic & knowledge.

I. INTRODUCTION

Phishing is a model problem for illustrating usability concerns of privacy & security because both system designers & attackers battle using user interfaces to guide (or misguide) users. Careful analysis of the phishing problem promises to shed light on a wide range of security usability problems.

In this paper, we studied the case of users authenticating web sites in the context of phishing attacks. In a phishing attack, the attacker spoofs a website (e.g., a financial services website). The attacker draws a victim to the rogue website, sometimes by embedding a link in email & encouraging the user to click on the link. The rogue website usually looks exactly like a known website, sharing logos & images, but the rogue website serves only to capture the user's personal information. Many phishing attacks seek to gain credit card information, account numbers, usernames & passwords that enable the attacker to perpetrate fraud & identity theft.

Data suggest that there were at least 55,698 phishing attacksin October2009. An "attack" is defined as a phishing site that targets a specific brand or entity. One domain name can host several discrete attacks against different banks, for example. This is down insignificantly from the 56,959 attacks recorded in 2008.Some phishing attacks have convinced up to 5% of their recipients to provide sensitive information to spoofed websites. About two million users gave information to spoofed websites resulting in direct losses of $1.2 billion for U.S. banks & card issuers in 2003. 2780 unique active phishing attack websites were reported in the month of March 2005 alone.

It is a dreary commentary on the state of Internet security that phishers are able to be so successful using straightforward attacks with little effort. Though we have known about spoofing vulnerabilities in browsers for years, some browser designers initially believed that these vulnerabilities were only an academic concern that deserved little attention. However, as we depend more on the Internet to conduct business & e-commerce transactions, the need to address spoofing vulnerabilities becomes more important.

The phishing problem shows that we as security designers have a distance to travel. Because both attackers & designers use user interface tools, examining this problem yields insight into usability design for other privacy & security areas.

II. SECURITY PROPERTIES

Why is security design for phishing hard? Variety of system proposed to thwart phishing; but these systems appear to be of limited success. Here are some properties that come into play.

A. The Limited Human Skills Property

Humans are not general purpose computers. They are limited by their inherent skills & abilities. This point appears obvious, but it implies a different approach to the design of security systems. Rather than only approaching a problem from a traditional cryptography-based security framework (e.g., "what can we secure?"), a usable design must take into account what humans do well & what they do not do well. As an example, people often learn to screen out commonly reoccurring notices. Browsers often warn users when they submit form data over an unencrypted connection. This warning is so common that most users ignore it, & some turn the warning off entirely.

B. The General Purpose Graphics Property

Operating systems & windowing platforms that permit general purpose graphics also permit spoofing. The implications of this property are important: if we are building a system that is designed to resist spoofing we

BIOINFO Security Informatics
Volume 1, Issue 1, 2011, pp-01-05
Available online at: http://www.bioinfo.in/contents.php?id=344

must assume that uniform graphic designs can be easily copied.

C. The Golden Arches Property

Organizations invest a great deal to strengthen their brand recognition & to evoke trust in those brands by consumers. Just as the phrase "golden arches" is evocative of a particular restaurant chain, so are distinct logos used by banks, financial organizations, & other entities storing personal data. Because of the massive investment in advertising designed to strengthen this connection, we must go to extraordinary lengths to prevent people from automatically assigning trust based on logos alone. This principle applies to the design of security indicators & icons as well. For example, users often implicitly place trust in security icons (such as the SSL closed lock icon), whether they are legitimate or not.

D. The Unmotivated User Property

Security is usually a secondary goal. Most users prefer to focus on their primary tasks, & therefore designers can not expect users to be highly motivated to manage their security. For example, we can not assume that users will take the time to inspect a website certificate & learn how to interpret it in order to protect themselves from rogue websites.

E. The Barn Door Property

Once a secret has been left unprotected, even for a short time, there is no way to guarantee that it can not been exploited by an attacker. To be fully effective, anti-phishing solutions must be designed with these properties in mind.

III. INFORMATION ANALYSIS

The Anti Phishing Working Group [APWG] maintains a "Phishing Archive" describing phishing attacks dating back to September 2003. Reviewing these reports, we constructed a task analysis of the methods & necessary skills for a user to detect a phishing attack. Space limitations prevent us from presenting the full task analysis here; it is available in a companion report. Here we summarize our findings. We find that all of the attacks exploit the human tendency to trust certain brands, logos & other trust indicators. These attacks often ironically exploit a widespread sense that the Internet is unsafe & that users must take active steps to "protect" their financial accounts & passwords. The similarity between phishing attacks, which claim that users must update passwords, account activity, etc., & legitimate security requests adds verisimilitude to phishing attacks. The efficacy of phishing attacks is diminished when users can not reliably distinguish & verify authoritative security indicators. Unfortunately, current browser & related application programs have not been carefully designed with "security usability" in mind. As a result, users have the following problems.

Users can not reliably correctly determine sender identity in email messages. The email sender address is often forged in phishing attacks. Most users do not have the skills to distinguish forged headers from legitimate headers using today's email clients. Users can not reliably distinguish legitimate email & website content from illegitimate content that has the same "look & feel". If images & logos are mimicked perfectly, sometimes the only cues that are available to the user are the tone of the language, misspellings or the simple fact that large amounts of personal information is being requested. Users can not reliably parse domain names. Often they are fooled by the syntax of a domain name through "typejacking" attacks, which substitute letters that may go unnoticed (e.g. www.paypai.com & www.paypal.com), or when numerical IP addresses are used instead of text. The semantics of a domain name can also confuse users. (e.g., users can mistake www.ebaymembers-security.com as belonging to www.ebay.com). Legitimate organizations heighten this confusion by using non-standard naming strategies themselves (e.g., Citibank legitimately uses ci ti..com ,citicard.com & accountonline.com). Phishers have also exploited browser vulnerabilities to spoof domain names, for example by taking advantage of non-printing characters & non-ascii Unicode characters. Users can not reliably distinguish actual hyperlinks from images of hyperlinks. One common technique used by phishers is to display an image of a legitimate hyperlink. When clicked, the image itself serves as a hyperlink to a different rogue site. Even if the actual hyperlink is displayed in the status bar or a browser or email client, many users do not notice it.

Users can not reliably distinguish browser chrome from web page content. Browser "chrome" refers to the interface constructed by the browser around a web page (e.g., toolbars, windows, address bar, status bar). It is hard for users to distinguish an image of a window in the content of a webpage from an actual browser window. This technique has been used to spoof password dialogue windows, for example. Because the spoofed image looks exactly like a real window, a user can be fooled unless he tries to move or resize the window. Users can not reliably distinguish actual security indicators from images of those indicators. Many users can confuse a legitimate SSL closed-lock icon, which appears on the status bar, with an image of that icon in the content of a web page. Many users simply scan for the presence of a lock icon, regardless of where it appears. Furthermore, it is hard to train users exactly where to look, because each browser uses a different icon that appears in a different location of the browser chrome. Legitimate organizations heighten this confusion by allowing users to login from non-HTTPS

pages. A form POST from a HTTP page can be delivered securely via SSL, however, there is no visual cue to indicate if the data is sent via SSL or even to the correct server (e.g., Bank of America allows users to login from its HTTP homepage. Because the page itself is not SSL protected, the bank uses an image of a lock icon near the form to indicate that it is secure). Users do not understand the meaning of the SSL lock icon. Even if users can reliably identify a legitimate SSL lock icon on the status bar, they may be confused by what that icon actually means. The lock icon indicates that the page the user is viewing was delivered to the user securely. However, it does not guarantee that data entered into that page will also be sent securely to the server (e.g., a form on a HTTPS page may submit data to a non-HTTPS site). Some browsers provide warnings to inform the user when data is submitted insecurely, but many users ignore these warnings or turn them off. Users do not reliably notice the absence of a security indicator. In the Firefox browser, SSL protected pages are denoted by four indicators (a closed lock icon in the status bar, text of the actual domain name in the status bar, a closed lock icon in the address bar & a yellow background in the address bar). However, in the case of non-SSL protected web pages, each of these indicators is missing. Many users do not notice the absence of an indicator, & it is trivial to insert a spoofed image of that indicator where one does not exist. Users can not reliably distinguish multiple windows & their attributes. Users do not reliably understand SSL certificates. Very few users go through the effort of checking SSL certificates, & if they do, most do not have the skills to understand the information presented.

IV. Dynamic Secutity Skins

With the security properties & information analysis in mind, the predicted system should provide an authentication scheme that does not impose undue burden on the user, in terms of effort or time.

Dynamic Security skins provide following properties.

- To authenticate himself, the user has to recognize only one image & remember one low entropy password, no matter how many servers he wishes to interact with.
- To authenticate content from a server, the user only needs to perform one visual matching operation to compare two images.
- It is hard for an attacker to spoof the indicators of a successful authentication.

DSS use an SRP authentication protocol to achieve the following security properties: At the end of an interaction, the server authenticates the user, & the user authenticates the server.

- No personally identifiable information is sent over the network.
- An attacker can not masquerade as the user or the server, even after observing any number of successful authentications.

In this section, we provide an overview of solution in depth.

First, DSS provides the user with a trusted *password window*. This is a dedicated window for the user to enter usernames & passwords & for the browser to display security information. DSS present a technique to establish a trusted path between the user & this window that requires the user to recognize a photographic image.

Next, DSS present a technique for a user to distinguish authenticated web pages from “insecure” or “spoofed” web pages. DSS does not require the user to recognize a static security indicator or a secret shared with the server. Instead, the remote server generates an abstract image that is unique for each user & each transaction. This image is used to create a “skin”, which customizes the appearance of the server’s web page. The browser computes the image that it expects to receive from the server & displays it in the user’s trusted window. To authenticate content from the server, the user can visually verify that the images match.

DSS includes Verifier based Secure Remote *Password Protocol* (SRP), a developed by Tom Wu, to achieve mutual authentication of the user & the server. The use of SRP is because it aligns well with users’ preference for easy to memorize passwords, & it also does not require passwords to be sent over the network. The adaption of SRP protocol, allows the user & the server to independently generate the abstract skins.

V. Trusted Path to the Password Window

How can a user trust the client display when every user interface element in that display can be spoofed? DSS is solution in which the user shares a secret with the display, one that can not be known or predicted by any third party. To create a trusted path between the user & the display, the display must first prove to the user that it knows this secret. DSS’s approach is based on window customization [16]. If user interface elements are customized in a way that is recognizable to the user but very difficult to predict by others, attackers can not mimic those aspects that are unknown to them.

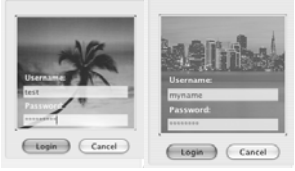

Fig. 1: The Trusted Password Window Uses a Background Image to Prevent Spoofing of the Window & Textboxes.

DSS provides the user with a *trusted password window* that is dedicated to password entry & display of

security information. It establishes a trusted path to this window by assigning each user a random photographic image that will always appear in that window. We refer to this as the user's *personal image*. The user should easily be able to recognize the personal image & should only enter his password when this image is displayed. As shown in Figure 1, the personal image serves as the background of the window. The personal image is also transparently overlaid onto the textboxes. This ensures that user focus is on the image at the point of text entry & makes it more difficult to spoof the password entry boxes (e.g., by using a pop-up window over that area).

VI. SECURE REMOTE PASSWORD PROTOCOL VERIFIER BASED PROTOCOLS

It is well known that users have difficulty in remembering secure passwords. Users choose passwords that are meaningful & memorable & that as a result, tend to be "low entropy" or predictable. In this authentication prototype, the goal is to achieve authentication of the user & the server, without significantly altering user password behavior or increasing user memory burden. We chose to implement a verifier-based protocol. These protocols differ from conventional shared-secret authentication protocols in that they do not require two parties to share a secret password to authenticate each other. Instead, the user chooses a secret password & then applies a one-way function to that secret to generate a verifier, which is exchanged once with the other party. After the first exchange, the user & the server must only engage in a series of steps that prove to each other that they hold the verifier, without needing to reveal it.

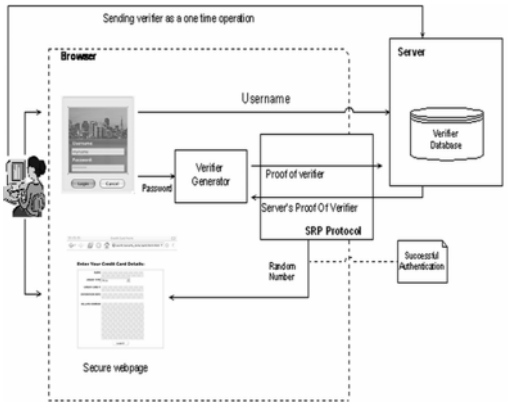

Fig. 2: The Actual Working of SRP in DSS.

SRP allows a user & server to authenticate each other over an untrusted network. SRP allows us to preserve the familiar use of passwords, without requiring the user to send his password to the server. Furthermore, it does not require the user (or his browser) to store or manage any keys. The only secret that must be available to the browser is the user's password (which can be memorized by the user & can be low entropy). The protocol resists dictionary attacks on the verifier from both passive & active attackers, which allows users to use weak passwords safely.

Here, we present a simple overview of the protocol to give an intuition for how it works. To begin, Carol chooses a password, picks a random salt, & applies a one-way function to the password to generate a verifier. She sends this verifier & the salt to the server as a one-time operation. The server will store the verifier as Carol's "password". To login to the server, the only data that she needs to provide is her username, & the server will look up her salt & verifier. Next, Carol's client sends a random value to the server chosen by her client. The server in turn sends Carol its own random values. Each party, using their knowledge of the verifier & the random values, can reach the same session key, a common value that is never shared. Carol sends a proof to the server that she knows the session key (this proof consists of a hash of the session key & the random values exchanged earlier). In the last step, the server sends its proof to Carol (this proof consists of a hash of the session key with Carol's proof & the random values generated earlier). At the end of this interaction, Carol is able to prove to the server that she knows the password without revealing it. Similarly, the server is able to prove that it holds the verifier without revealing it. The protocol is simple to implement & fast. Furthermore, it does not require significant computational burden, especially on the client end. Instead of forcing the user to remember many passwords, the browser can use a single password to generate a custom verifier for every remote server. This reduces memory requirements on the user; however it also increases the value of this password to attackers.

VII. CONCLUSION

In this paper, we studied that DSS provide more secure way to prevent phishing. Because uses a random image which can become skin to user browser. We studied SRP, which create mutual authentication between client & server which is more effective & secure than a SSL. This is due to the fact that The Image is generated from the session key as a result of SRP algorithm which is tougher than SSL based approach. This is because SRP mechanism is similar to Diffie Helman mathematical structure which is used to create random session key. At the last the user only has to submit only username, not password to server. After processing user authenticate server is legitimate, similarly server authenticate client is legitimate, without reveling any sensitive information over untrusted network.

VIII. REFERENCES

[1] APWG Phishing & eCrime News, http://APWG Phishing News.htm, Jan 27, 2010 01:55AM EST
[2] Anti-Phishing Working Group, Phishing Activity Trends Report March 2005, http://antiphishing.org/APWG_Phishing_Activity_Report_March_2005.pdf
[3] Tom Wu. Secure Remote Password Protocol. in the Stanford University.
[4] S. Bellovin,\Problem Areas for the IP Security Protocols", Proceedings of the Sixth USENIX Security Symposium, Usenix Association, 1996, pp.205{214.ftp://ftp.research.att.com/dist/smb/badesp.ps.
[5] A. Freier, P. Karlton, & P. Kocher,\The SSL ProtocolVersion 3.0",ftp://ftp.netscape.com/pub/review/ssl-spec.tar.Z, March 4 1996.
[6] Wombat Security Technologies, Phil, PhishPatrol & PhishGuru Technologies.
[7] Varjonen, RFC 2944 & 2945 (SRP Protocol), Helsinki Institute for Information Technology, February 28-2009